**General Purpose I/O**

For RLP-VLSI Research at UVA

**Current Researcher:** Ashley Morse, 3rd Year Undergrad (Spring 2014 )

**Current Advisor:** Alicia Klinefelter, Graduate (Spring 2014 )

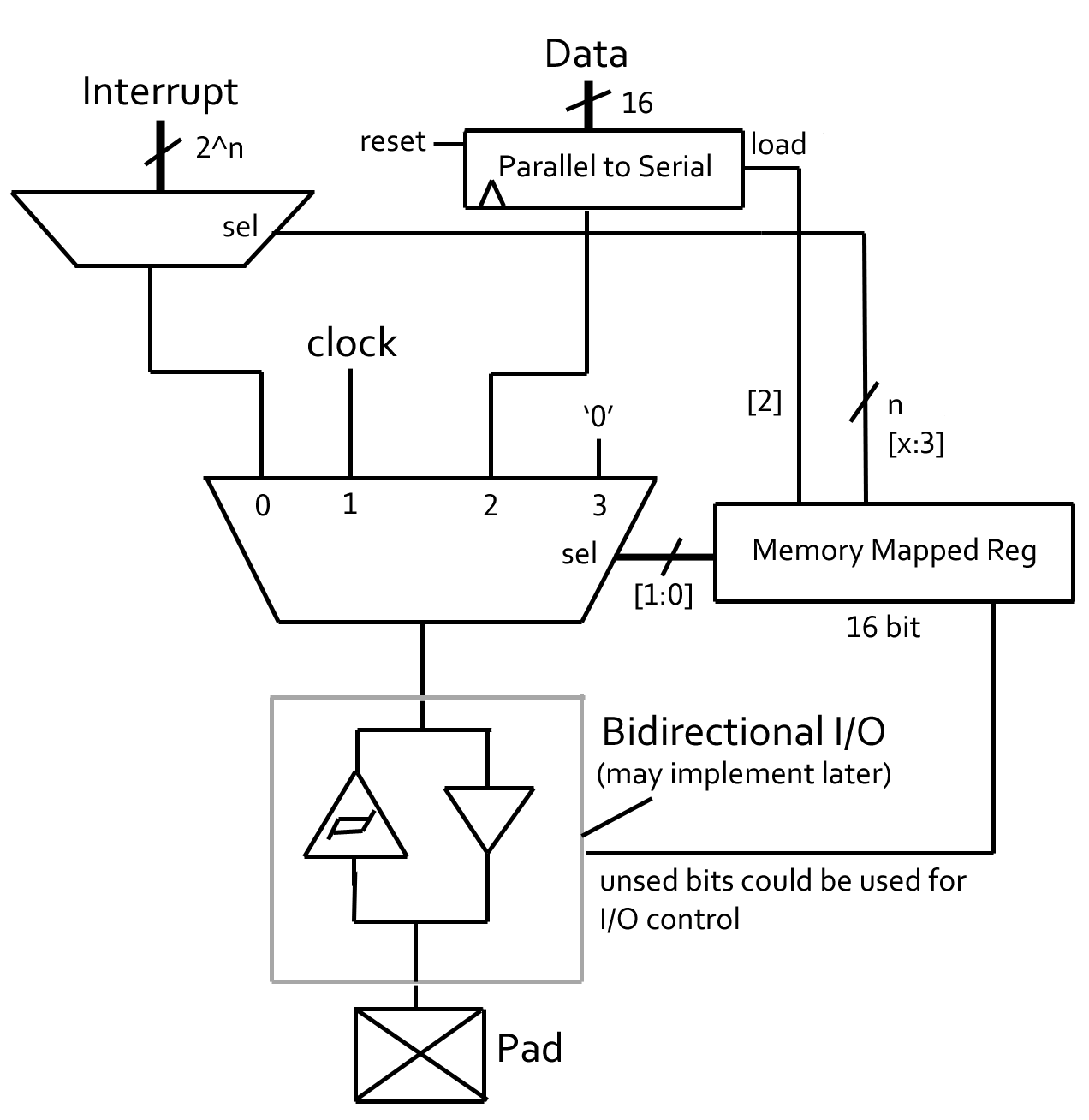
**Overview:**

To help improve testing in chips, a general purpose I/O (GPIO) component can be connected to the pad and use a mux to select certain data that we want to test. Area is saved this way, because more than one output can share one pad, which is in limited supplies on a chip. A bidirectional I/O section could be added between the GPIO and pad to conduct further testing, but that is not a top priority. The approach used is to understand the high level design, coding each section of the GPIO in Verilog/System Verilog, make sure it is synthesizable, and finally integrate it with the BSN architecture.

**Walkthrough**

1. **High Level Designing** 
   1. **GPIO Mux**

* The main section of the GPIO is a 4:1 1bit MUX, and it will have the ability to select from an Interrupt signal, clock, incoming bus data, and a zero signal. It’s select is a 16bit array of data that is memory mapped to a register that is used in other Verilog code.
  1. **Interrupt Mux**
* This mux will take in 2^n inputs, where n is the bit width of the select input. The mux will determine if interrupt signals are high throughout the circuit. Currently, the Verilog code of this mux has defined n for simplicity, but it is expected to be more flexible later.
  1. **Parallel to Serial Bus Data**
* 16bit Bus Data should go through a sequential block that outputs the least significant bit of the input. If load it ‘1’, the input will get loaded in, and while load is ‘0’, the output will equal the LSB of the input then shift it. There is also a ‘reset’ input that sets everything to 0 is set low.



*Figure 1- High level GPIO design*

1. **Verilog Code**

* The code related to the GPIO is called *gpio\_core.v*and can be found by contacting whoever is currently working on the chip.
* It is recommended that a test bench be created to ensure this woks correctly, and can be synthesizable.

**Next Steps**

* To build a testbench and test gpio\_core.v
* Integrate code with BSN architecture